

Um Modelo baseado em Padrões Web para Apresentação de Conteúdo Interativo em Meios Sociais

Marcio dos Santos Galli
Taboca.com - Taboca Labs
São Carlos, SP, Brasil
mgalli@taboca.com

Felipe Tassario Gomes
Taboca.com - Taboca Labs
Mountain View, CA, USA
felipe.gomes@gmail.com

RESUMO

A crescente disponibilidade de painéis digitais oferece novas oportunidades com relação as soluções de apresentação de conteúdo nos ambientes físicos nas comunidades. Com as melhorias de conectividade na Internet e a constante evolução das tecnologias que dão suporte aos padrões Web, novos desafios de interação usuário-computador podem ser explorados. Nesse contexto, o trabalho aborda as possibilidades de uso de padrões e tecnologias Web, como infra-estrutura para uma experiência de comunicação e interação em ambientes físicos. Propõe-se um modelo de programação que seja aberto, distribuído, e que reflète a natureza dinâmica da Web, em contraposição ao modelo centralizado de programação tradicional, como da televisão. O trabalho elabora sobre características de uma aplicação que permite levar esta experiência de conteúdo, visando benefícios para usuários nas comunidades.

Palavras-chave

Web 2, feeds, social-aware computing, W3C, RSS, DOM, JavaScript, mashup, computação pervasiva, computação ciente de contexto

1. INTRODUÇÃO

A utilização de painéis digitais é cada vez maior em ambientes públicos e semi-públicos, como escolas, terminais de transporte, entre outros. Existem soluções de software, referidas como *Digital Signage* [11], que são criadas para prover experiências de apresentação visual em tempo real — seja este conteúdo presente no computador local, na intranet, ou mesmo armazenado remotamente na Web. Com o crescente investimento da indústria de painéis, como monitores de LED ou LCD, existe uma redução de custos de equipamentos, e melhorias em relação a qualidade em aspectos como resolução, brilho, contraste e até durabilidade de materiais. Hoje, a utilização de painéis tipo LED, permite uma experiência de apresentação visual na luz do dia, além de oferecer maior economia de energia.

Este cenário cria oportunidades quanto aos paradigmas de visualização e interação usuário-computador, pois difere de experiências do tipo *desktop* ou de dispositivos móveis. Neste sentido, o trabalho busca contribuir com uma infra-estrutura, baseada em padrões abertos, que permite a governança de painéis que façam interação com conteúdo da Web e possam contribuir com o meio local.

O sistema proposto é baseado na tecnologia de um navegador Web e funciona como um mediador entre canais de

conteúdo Web e usuários participantes no local. Contrapondo os tradicionais sistemas de televisão, que têm uma programação centralizada, este sugere cenários que promovem uma experiência de programação de conteúdo distribuída com base em eventos que ocorrem na Web — dados oriundos de canais disponíveis via HTTP. Nesse sentido, busca-se preservar os valores da Internet, como por exemplo a capacidade de escolha quanto aos canais de informações, criando-se oportunidades para experiências que são atreladas ao interesse local e benefícios para a comunidade.

É proposto um sistema, denominado de navegador social, que com base na infra-estrutura de padrões Web, forneça uma experiência de acesso e interação com o conteúdo Web e que seja disponível através de painéis digitais em ambientes. Com isto, busca-se permitir a implantação de experiências de comunicação nas comunidades, assim como prover uma infra-estrutura inicial para a captura de dados com o objetivo de permitir trabalhos de análise sobre os casos de uso.

2. PROJETO NAVEGADOR SOCIAL

Esta seção apresenta a arquitetura geral do sistema. Em linhas gerais, o termo navegador social é proposto como um sistema que coordena a apresentação visual de conteúdo Web em um dado ambiente físico com usuários. O sistema possibilita a execução de cenários, oferecendo uma experiência de programação de conteúdo descentralizada. A figura 1 destaca o espaço dos eventos Web, como as fontes de conteúdo de sítios e o espaço de acontecimentos de interesse de uma comunidade. A interseção representa um interesse neste trabalho, ou seja, quais eventos do meio local podem coordenar a governança de uma experiência que faz interação com conteúdo externo disponível em sítios, como por exemplo redes sociais.

Tendo por base este espaço, busca-se oferecer uma experiência de conteúdo que ocorre através da reutilização de recursos disponíveis na Web, como os componentes do tipo *widgets* [13] e conteúdo disponível em formatos de *feeds* [3], como RSS [10]. Assim, permite-se que experiências do tipo *mashup* [8] possam ser apresentadas nos painéis. Neste modelo, existe uma separação entre o aspecto visual e o conteúdo da Web, que é proposta através de um modelo de eventos, que permite o relacionamento entre os *widgets* de apresentação e os eventos de conteúdo.

Serão apresentadas capacidades que suportam o desenvolvimento de uma aplicação de software, denominado TelaSocial, que é baseada em padrões Web e na plataforma Mozilla XULRunner [14]. Em linhas gerais, a arquitetura introduz funcionalidades encontradas em leitores de *feeds*, como por

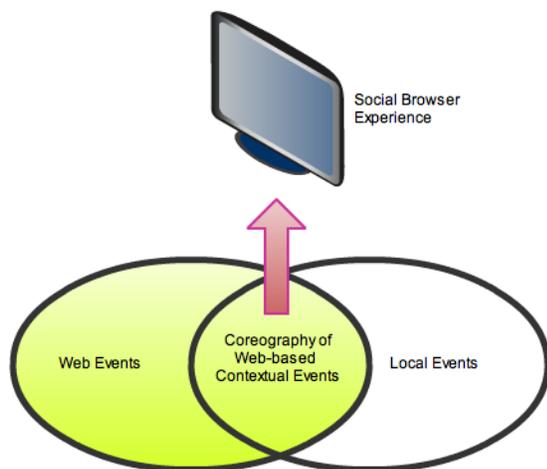


Figura 1: Intersecção destaca a área interesse, com eventos da Web e da localidade

exemplo a capacidade de armazenamento dos *feeds* em base local, e introduz um modelo que dá suporte para a governança do conteúdo e que permite uma experiência contínua, por exemplo em ambientes públicos e semi-públicos.

3. ASPECTOS DA INFRA-ESTRUTURA

3.1 Toolkit de Interação Web

O sistema proposto incorpora mecanismos de um navegador Web e funcionalidades de um leitor agregador de *feeds*. Esta aplicação, que tem base no Mozilla XULRunner, está disponibilizada com licença de código aberto, através do projeto TelaSocial [12]. Nesta seção, o termo *toolkit* refere-se a infra-estrutura de várias tecnologias e funcionalidades fundamentais encontradas nos navegadores e que são baseadas em padrões abertos Web. Esta infra-estrutura inclui, por exemplo, a funcionalidade que permite o acesso as informações de conteúdo do tipo *feeds* via protocolo HTTP.

Também incluem-se as capacidades de execução de conteúdo Web, que permitem que aplicações de padrões como XHTML, SVG, CSS, DOM, PNG, OGG possam ser apresentadas para usuários. A figura 2 apresenta a arquitetura *Gecko*, que é a camada fundamental da tecnologia Mozilla, e ilustra várias funções de suporte quanto aos padrões abertos. O protótipo, TelaSocial, situa-se em cima, na camada de aplicação ou extensão, e assim, utiliza e disponibiliza estas tecnologias Web, além de várias outras funcionalidades que são importantes quanto as sessões de navegação.

3.2 Suporte Web Offline

Dada uma sessão navegação, onde um usuário utiliza-se de um navegador para o acesso a várias páginas, existem funções de persistência de dados que permitem uma experiência rica entre sessões de navegação. Como por exemplo, as funções de histórico ou *bookmarks* permitem que um usuário possa revisitar uma certa página. Neste sentido, o HTML 5 estende este modelo para o nível de aplicações Web, permitindo que páginas possam armazenar dados entre sessões. Esta funcionalidade é definida pela especificação Offline Webapps [9]. No caso de uma aplicação do tipo Web Mail, o

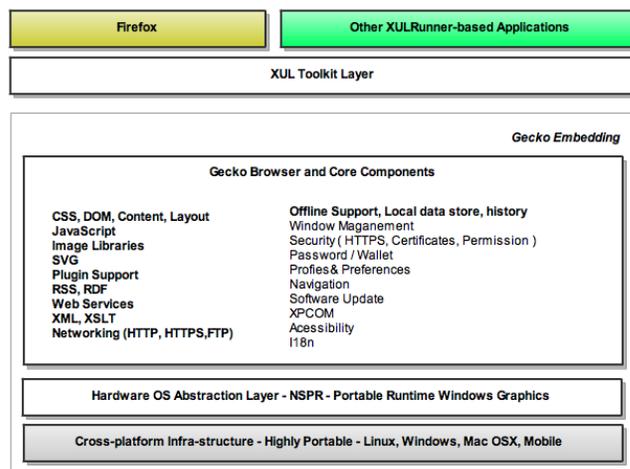


Figura 2: Arquitetura Gecko em camadas

suporte *offline* pode permitir a implementação de um comportamento onde o usuário possa compor uma mensagem nova, voltar ao *inbox*, ler outras mensagens, tudo isto mesmo que a conexão com a Web esteja temporariamente fora do ar.

No contexto do projeto, uma infra-estrutura de API *offline* é critério fundamental, pois permite a implementação de algumas capacidades do sistema. Uma delas é que dados de conteúdo possam ser agregados no contexto local da aplicação cliente, permitindo uma posterior apresentação visual e interação baseada em vários fatores, como por exemplo, agenda. Outro critério, que garante qualidade do sistema, trata da capacidade de se manter referência dos eventos que ocorrem no sistema de tal forma que, se a conexão com a Internet cair, o sistema mantém uma experiência de comunicação persistente.

4. CAPACIDADES E CENÁRIOS

4.1 Canais de Conteúdo e Eventos de Feed

Na navegação tradicional, controlada por um usuário, o acesso às páginas ocorre através de eventos de entrada. Neste caso, a autonomia do controle da navegação é do usuário, que ao clicar em *links*, gera eventos que fazem o navegador obter novas páginas. Por outro lado, no modelo do navegador social, sugere-se que a interação com recursos Web se faça através de eventos de entrada que não são diretamente associados com as ações de usuários. Para este fim, propõe-se a utilização de *feeds*.

Em geral, sítios de notícias disponibilizam a coleção dos últimos artigos via a utilização de canais de *feeds*, por exemplo utilizando-se do formato RSS, entre outros. Estes formatos são bem suportados por vários sistemas de servidores Web.

Uma das capacidades propostas é que os itens de *feeds* sejam processados individualmente uma vez que o conteúdo de um dado canal de *feed* é recuperado pelo sistema, tornando-se eventos de *feed*. Estes eventos serão posteriormente tratados por regras, que definirão as relações de coreografia, abordadas na próxima seção.

A figura 3 apresenta um documento em formato RSS, ao

lado esquerdo, dado um canal de *feed*. Ao lado direito, uma base de dados é mantida, com o objetivo de caracterizar, de forma individualizada, os itens de *feed* do canal.

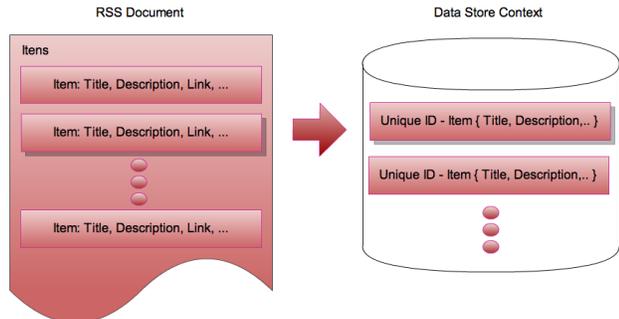


Figura 3: Eventos de *feed* criados a partir de um canal de *feed*

Uma vez que cada item do *feed* é processado, este passa a ser inserido como evento de *feed* em uma fila de eventos, que é representada em uma estrutura de objetos do tipo DOM [2]. Este modelo facilita a portabilidade destes eventos e posteriormente, através do uso de regras, serão definidos aspectos sobre a formação de contextos e ações no sistema. Estes itens de *feed* são mantidos com o suporte da infra-estrutura *offline*, que além de manter dados persistentes durante sessões, ainda garante a experiência de eventos até em casos como queda da Internet ou indisponibilidade dos canais. O código abaixo é um exemplo de um item de *feed* que foi extraído de um documento RSS e passa a existir individualmente. Destaca-se o elemento `feeditem:event`, que contém a referência para o item de *feed* originalmente encontrado no documento RSS, além de possuir alguns atributos adicionais que servem como anotação de contexto sobre o canal original.

```
<feeditem:event id="1" domain="usp.br" path="/noticia">
  <item><title /><description /><link /> ... </item>
</feeditem:event>
```

4.2 Regras de Coreografia

A contextualização dos eventos de *feed* é um aspecto fundamental, pois permite que a existência de itens originados a partir de RSS, possam gerar, por exemplo, experiências de animação para usuários. Para isto, é proposto um modelo de regras, que definirão aspectos sobre a coordenação dos eventos em uma sessão de execução. Estas regras permitem que novos eventos, referidos aqui como eventos de contexto, sejam gerados em condições do conjunto de eventos de *feed*. Para ilustrar um caso de uso, tem-se elementos de *feed* inicialmente obtidos por dois canais de RSS, por exemplo `http://www.slashdot.org (slashdot)` e `http://www.w3.org (w3)`. Um possível interesse, é a manutenção de um evento denominado "Artigo", que acontece sempre que existem itens de *feed* do sítio slashdot ou itens do sítio w3. Este caso de uso é comum em aplicações do tipo leitores de *feed*, uma vez que usuários tem interesse em ler notícias de vários sítios, por exemplo através de uma categorização por data.

Neste modelo, é proposta a utilização de seletores, que é parte da especificação CSS [7]. O uso de seletores permite a especificação de regras de filtro que identificam elementos. Desta forma, é possível fazer a seleção dos eventos de *feed*

uma vez que estes se encontram disponíveis em uma abstração de dados disponível em um documento tipo DOM. A estrutura de regra, baseada nos seletores do CSS, permite a especificação de uma seleção e a definição de propriedades que são atribuídas aos elementos da seleção:

```
selector {property: value}
```

Neste modelo, a semelhança com CSS é limitada ao contexto do seletores, pois nas regras de coreografia, sugere-se um tipo de propriedade que dá suporte para contextualização e acesso aos itens da seleção. Esta propriedade é denominada `-context-event`, e permite formar um novo evento no sistema, um evento de contexto aos itens da seleção.

No exemplo abaixo, observa-se regras que ditam a criação de eventos com nome Artigo, que conterà a referência aos elementos de *feed* limitados pelo seletor. Neste modelo, quando a primeira regra é satisfeita, um estado de contexto, denominado Artigo, é criado e mantido em memória. Uma vez que este estado existe, um evento Artigo é também gerado em uma fila de eventos de contexto. Quando a segunda regra, referente a `w3.org`, é satisfeita, o estado de contexto Artigo é atualizado e passa a receber também a referência dos elementos estabelecidos nesta segunda regra, e um novo evento Artigo é enviado na fila de eventos de contexto. Com isso pode-se implementar um sistema de notificação sempre que ocorrem novos artigos, itens de *feed*, de ambas as fontes de conteúdo, além de se manter a referência para os elementos, os itens.

```
feeditem | event [domain="www.slashdot.com"] {
  -context-event: Artigo;
}

feeditem | event [domain="www.w3.org"] {
  -context-event: Artigo;
}
```

Desta maneira, toda vez que um evento de *feed* entra na fila de eventos, as regras de coreografia são verificadas e poderão gerar os eventos de contexto. No exemplo, uma vez que cada regra é satisfeita, a abstração de dados Artigo, através do estado de contexto, contém referências para os conjuntos de elementos gerados pela primeira e segunda regras. A existência do evento é importante no sentido do fluxo de informações no tempo, mas o estado de contexto funciona como um uma chave ou índice que permite acesso direto aos elementos das seleções referidas.

4.3 Composição com Eventos de Contexto

Uma das capacidades do projeto é possibilitar o gerenciamento de relacionamentos e operações com eventos de contexto. Um estado de composição é definido como uma abstração que identifica um evento de contexto que se relaciona com outros eventos de contexto que ocorreram anteriormente. Posteriormente, estes estados serão utilizados para determinar ações no âmbito dos componentes Web, permitindo assim, uma apresentação e efeitos visuais.

O modelo de composição [5] da figura 4 apresenta um exemplo que descreve relacionamentos entre os eventos que ocorrem em uma sessão. A árvore apresenta uma composição na qual o evento estado zoom-menu passa a existir devido a um relacionamento a partir dos estados de contexto nos níveis anteriores, abaixo, na árvore. A partir deste modelo, nota-se que o evento estado show-menu seja criado,

caso os seus eventos relacionados existam anteriormente na sessão. Da mesma forma o zoom-menu deve ocorrer.

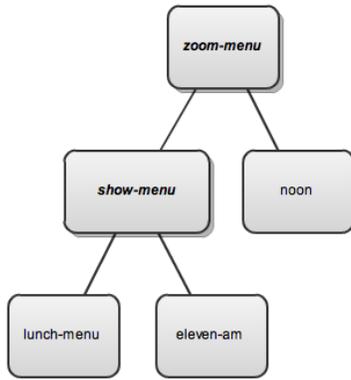


Figura 4: Destaque para os eventos de contexto que tem relacionamento

Deste modo, para que o contexto de relacionamento seja criado, é necessário que a regra de coreografia determine que os estados esperados estejam existentes na sessão. A regra abaixo descreve a criação do evento de contexto show-menu através do uso de um operador do tipo general-sibling-combinator, que é parte da recomendação CSS3 Selectors [15]. Este operador diz que ambos lunch-menu e eleven-am devem existir para que a regra seja válida.

```

lunch-menu ~ eleven-am {
    -context-event: show-menu;
}
  
```

Outros tipos de operações funcionais podem estabelecer estados de relacionamentos entre eventos. A figura 5 apresenta um exemplo, no qual um evento de contexto Arrange é formado sempre que quaisquer dos eventos A, B, C ou D acontecem. Quando o evento Arrange ocorre, uma função de relacionamento é chamada. No escopo desta função pode-se acessar os elementos que se relacionam com Arrange, abaixo na árvore, e assim pode-se por exemplo calcular a quantidade total de filhos de Arrange. Para que o evento SUM represente o valor soma, esta função precisa escrever atributos no estado de contexto do evento SUM. Desta maneira, um possível observador poderia verificar quantos elementos existem no contexto Arrange e assim fazer uma apresentação visual apropriada.

4.4 Videografia e Widgets de Apresentação

Na televisão e cinema, técnicas de videografia são utilizadas por equipes profissionais com o objetivo de criar experiências de observação. Captura de imagem e edição de filmes é um processo caro que envolve arte, técnicas de cinematografia, equipes e sistemas profissionais. O trabalho de videografia virtual [6], propõe a utilização de sistemas de software com o objetivo de se criar experiências de captura e edição de vídeo, o que dispensa equipes de filmagem e edição em certos cenários. Como por exemplo, dada a cena em vídeo de uma aula, o professor poderia ser removido da frente do quadro dependendo apenas de critérios de captura e observação. Em outros casos, o sistema pode ativar um efeito

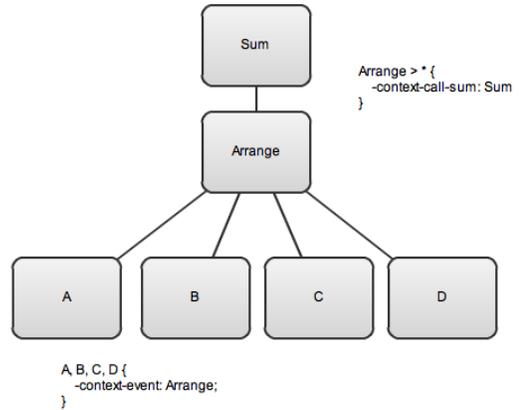


Figura 5: Existência de eventos A,B,C,D implicam que a operação do tipo soma seja calculada

de zoom criando-se por exemplo uma experiência visando destaque.

Esta ideia valoriza as intenções de observação em uma dada cena de acontecimentos. Além das intenções, as técnicas de edição visual permitem as execuções de efeitos no vídeo e assim, caracteriza-se a cena de observação. No contexto do navegador social, o equivalente a cena passa a ser coleção de eventos de entrada. Os eventos de contexto, originados pelas regras de coreografia, funcionam como indicadores e permitem que experiências de apresentação possam ser ativadas.

A utilização de *widgets* é proposta como modelo para definição dos componentes de apresentação baseados em padrões abertos. A infra-estrutura do *toolkit* Web permite que *widgets* possam ser dinamicamente inseridos ou removidos no contexto do documento de representação visual do sistema, que pode ser um documento tipo HTML. A interação entre os eventos de contexto e componentes *widgets* é proposta via eventos de ação que disparam a execução de funções no escopo dos *widgets*.

A figura 6 coloca um caso de uso com uma experiência visual de um calendário que possui duas funções principais. A primeira permite a atualização de conteúdo referente ao dia. A outra é a ação de se alternar a folha do mês. Assumindo-se que um novo item de calendário é representado pelo evento UpComing, um evento de ação permitirá que o *widget* do calendário possa se atualizar. Um outro evento de ação é associado pela regra FlipCalendar, que acontece na ocorrência de um evento de atualização de mês, MonthEvent.

```

UpComing {
    -event-action: calendar.update;
}
MonthEvent {
    -event-context: FlipCalendar
}
FlipCalendar {
    -event-action: calendar.flip
}
  
```

Os eventos de ação permitem a reutilização de *widgets* em contextos diferentes, e também o mapeamento entre os eventos de contexto com possíveis operações que são executadas no escopo do *widget*. Uma vez que o *widget* recebe

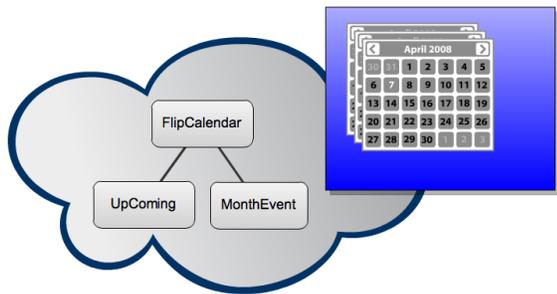


Figura 6: Coreografia de eventos visando comportamentos de interação visual

um evento, o estado de contexto do evento é passado ao *widget*, possibilitando o acesso aos elementos de conteúdo. Dependendo dos relacionamentos de coreografia, pode-se criar estados de contexto que agregam informações no tempo, e assim estabelecer vínculos de atualização no escopo do *widget*. Como por exemplo, pode-se acumular dados em um *widget* que mantém a visualização de um gráfico de pizza.

Em outro cenário de uso, é possível a utilização de efeitos visuais que podem ser aplicados a um ou vários *widgets* em uma página de visualização. Este efeito pode ser executado inserindo-se regras de estilo CSS em tempo de execução no documento de apresentação em HTML. De maneira similar aos eventos de ação, é necessário que um evento de ação seja definido, e que uma ação de geração de estilo seja executada. Neste caso, a característica principal é que as regras de estilo estariam sendo aplicadas aos elementos em tempo de execução. O código abaixo exemplifica uma regra de efeito de transição utilizando-se de CSS3 Transitions [1].

```
targetwidget {
  transition-property: zoom, left, top;
  transition-duration: 4s, 4s, 4s;
}
```

5. CONSIDERAÇÕES FINAIS

No modelo proposto, é sugerida a utilização de padrões abertos Web e tecnologias da infra-estrutura de um navegador Web, visando uma experiência de interação inicialmente para espaços físicos de convívio das pessoas. Neste sentido, abordou-se cenários que exploram um modelo de eventos para governança de conteúdo baseado em fontes de conteúdo de redes na Web, como por exemplo redes sociais.

Com este trabalho, busca-se contribuir com uma experiência de comunicação social que é compatível com a natureza evolutiva da Web, por exemplo, promovendo a reutilização de componentes visuais para fins de apresentação de conteúdo em ambientes públicos e semi públicos. Também busca-se explorar a criação de modelos de coreografia para representação de vários cenários de uso, visando causar oportunidades de interação e um modelo participativo para comunidades. Em casos de experimentação do protótipo inicial [4], foi observado interação de usuários através da disponibilização de experiências com redes sociais utilizando-se canais de *feeds*. Em um dos testes, o painel foi colocado em local tipo conferência, monitorando um canal de interesse disponível no sítio Twitter.com. Foi constatada a participação por usuários da localidade, através de ação indireta, via acesso ao Twitter.com por computadores da universidade.

Trata-se de um resultado subjetivo, pois existe a presença física do painel, além de *feedback* em tempo real, que são fatores motivadores. No entanto, entende-se como positivo pois tratou-se de participação relevante ao contexto da conferência local.

Em termos de continuidade, busca-se incluir melhorias quanto a captura da atividade do sistema, visando trabalhos de análises quanto aos cenários e modelos para governança e manutenção das regras de coreografia. Busca-se explorar novas possibilidades de interação, visando experiências dinâmicas com a comunidade, além de melhorias na aplicação, visando maiores projetos e potencialmente trabalhos aplicados nas comunidades. Maiores informações sobre o sistema podem ser encontradas no repositório:

<https://github.com/taboca/TelaSocial>

6. AGRADECIMENTOS

Agradecimentos à ICMC Jr Empresa de Alunos do ICMC USP em São Carlos, a Paulo Kinicky pelo trabalho de organização do projeto e documentação. Aos pesquisadores Edson Moreira e José Alberto Cuminato do ICMC, Luiz Carlos Dotta, Maria Alice Soares de Castro e Dagoberto Carvalio Junior que proporcionaram as oportunidades para implantação real e análise dos estudos de casos. Agradecimentos à Thiago José Campos e Rodrigo Pedra Brum pelas contribuições de tecnologia para o protótipo.

7. REFERÊNCIAS

- [1] D. H. Dean Jackson and C. Marrin. Css transitions module level 3 (working draft). Technical report, W3C, March 2009.
- [2] Document object model. <http://www.w3.org/DOM/>.
- [3] Web feeds. <http://en.wikipedia.org/wiki/Webfeed>.
- [4] M. Galli. Tela do icmc-usp. Technical report, Taboca Labs, Fevereiro 2010.
- [5] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns*, section 4, pages 163–166. Addison-Wesley, Boston, MA, January 1995.
- [6] R. Heck, M. Wallick, and M. Gleicher. Virtual videography. *ACM Trans. Multimedia Comput. Commun. Appl.*, 3(1):4, 2007.
- [7] H. W. Lie and B. Bos. Cascading style sheets, level 1. Technical report, W3C, April 2008.
- [8] Mashup - web application hybrid. [http://en.wikipedia.org/wiki/Mashup_\(web_application_hybrid\)](http://en.wikipedia.org/wiki/Mashup_(web_application_hybrid)).
- [9] Offline web applications. <http://www.w3.org/TR/offline-webapps/>.
- [10] What is rss. <http://www.xml.com/pub/a/2002/12/18/dive-into-xml.html>.
- [11] Digital signage. http://en.wikipedia.org/wiki/Digital_signage.
- [12] Sinalização social com tecnologia web. <http://www.telasocial.com/folder-uni.html>.
- [13] Widgets web. http://en.wikipedia.org/wiki/Web_widget.
- [14] Mozilla xulrunner. <https://wiki.mozilla.org/Xulrunner>.
- [15] T. Çelik, E. J. Etemad, D. Glazman, I. Hickson, P. Linss, and J. Williams. Selectors level 3 (working draft). Technical report, W3C, March 2009.